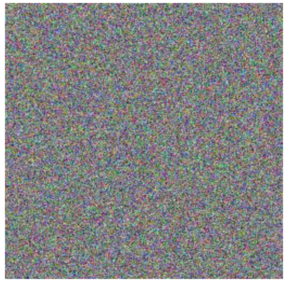
Introduction to Diffusion Models

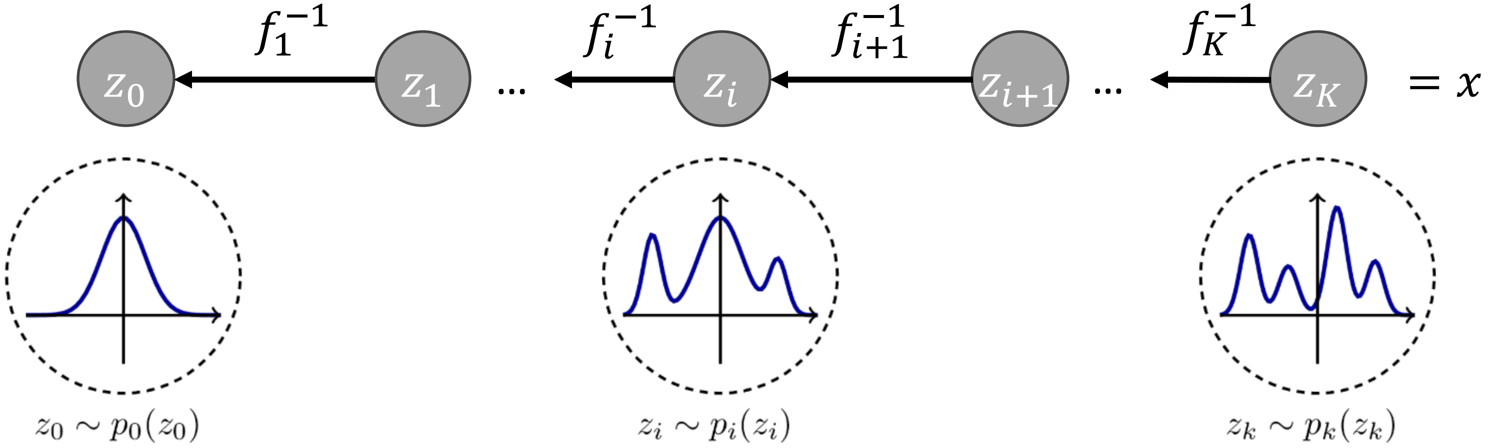
Lecture 8

18-789

Recap: Normalizing Flows

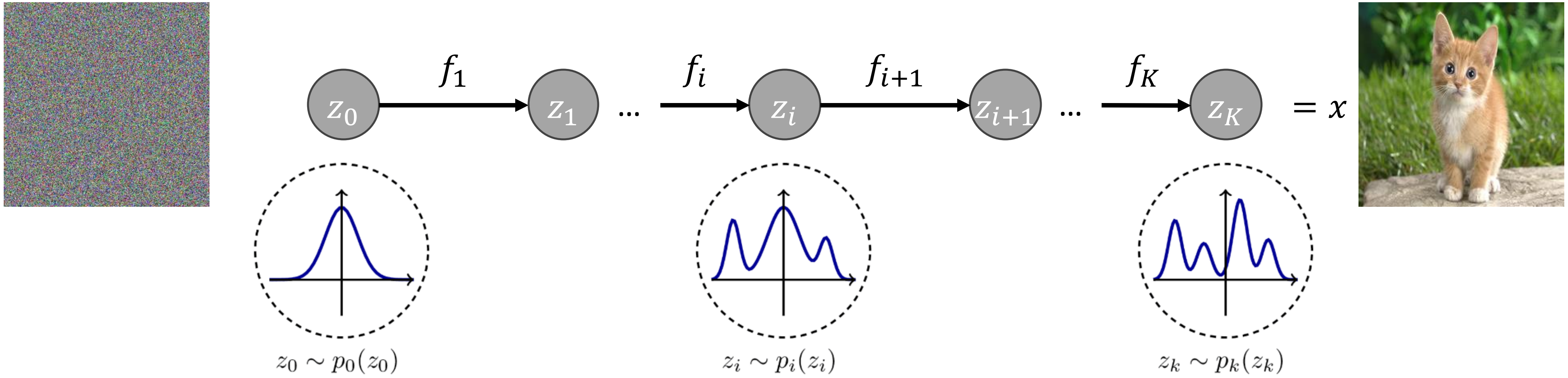
- Training: Maximum likelihood
  - **Encode** data to latent (Gaussian/**Normal** distribution)
  - Compute loss (negative log-likelihood) and backpropagate
    - $\log p_x(x;\theta) = \log p_z\left(f_\theta^{-1}(x)\right) - \sum_{i=1}^{K} \log\left|\det \frac{\partial f_i}{\partial z_{i-1}}\right|$





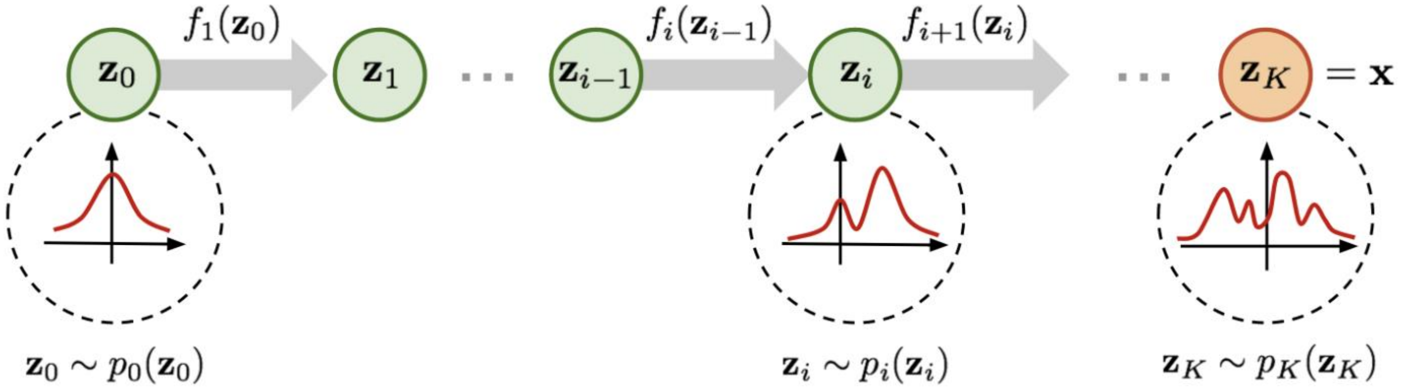
Recap: Normalizing Flows

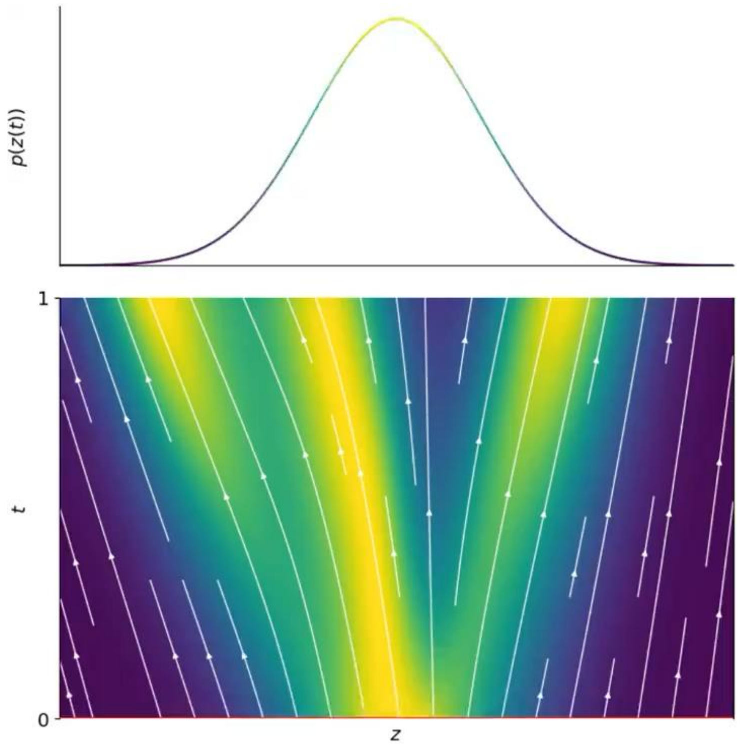
- Generation: iterative transformation
  - **Decode** data from latent (Gaussian/**Normal** distribution)



Continuous Normalizing Flows (CNFs)

- Normalizing Flows consist of K discrete transformations
  - $z_i = f_i(z_{i-1}),\ z_K = f_K \circ f_{K-1} \circ \cdots \circ f_2 \circ f_1(z_0)$
- Generalize to continuous case
  - **ODE**: $\frac{\partial z_t}{\partial t} = f(z_t, t), 0 < t < 1$
  - $z_{t+\nabla t} \approx f(z_t, t)\nabla t + z_t,\ z_1 = z_0 + \int_0^1 f(z_t, t)\, dt$

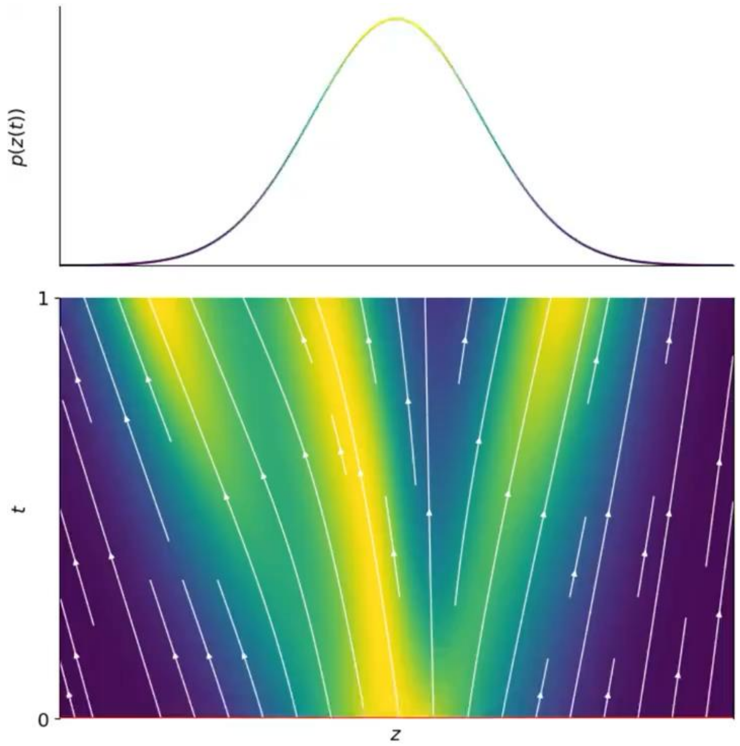




“Neural Ordinary Differential Equations”, Chen et al., 2018

Continuous Normalizing Flows (CNFs)

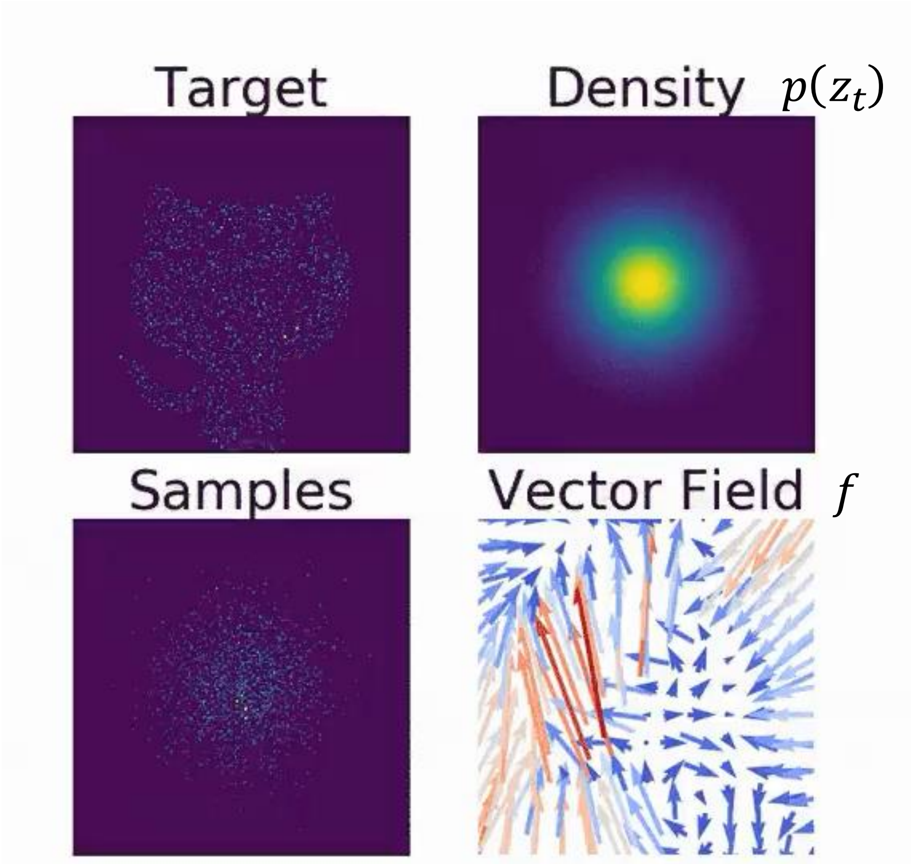
- Normalizing Flows consist of K discrete transformations
  - $z_i = f_i(z_{i-1}),\ z_K = f_K \circ f_{K-1} \circ \cdots \circ f_2 \circ f_1(z_0)$
- Generalize to continuous case
  - **ODE**: $\frac{\partial z_t}{\partial t} = f(z_t, t), 0 < t < 1$
  - $z_{t+\nabla t} \approx f(z_t, t)\nabla t + z_t,\ z_1 = z_0 + \int_0^1 f(z_t, t)\, dt$
- Training objective
  - $\log p(z_1) = \log p(z_0) - \int_0^1 \mathrm{Trace}\left(\frac{\partial f}{\partial z_t}\right) dt$
  - Assume z_0 is noise and z_1 is data



"Neural Ordinary Differential Equations", Chen et al., 2018

Continuous Normalizing Flows (CNFs)

- Network
  - A neural network $f(z_t, t)$ conditioned on data z_t and time t
  - Unrestricted architecture
- Training
  - $\log p(z_1) = \log p(z_0) - \int_0^1 \mathrm{Trace}\left(\frac{\partial f}{\partial z_t}\right) dt$
  - Solve the forward **ODE** to compute log-likelihood
  - Backpropagate through the **ODE**
- Sampling
  - Solve the backward **ODE**
  - $z_1 = z_0 + \int_0^1 f(z_t, t)\, dt$



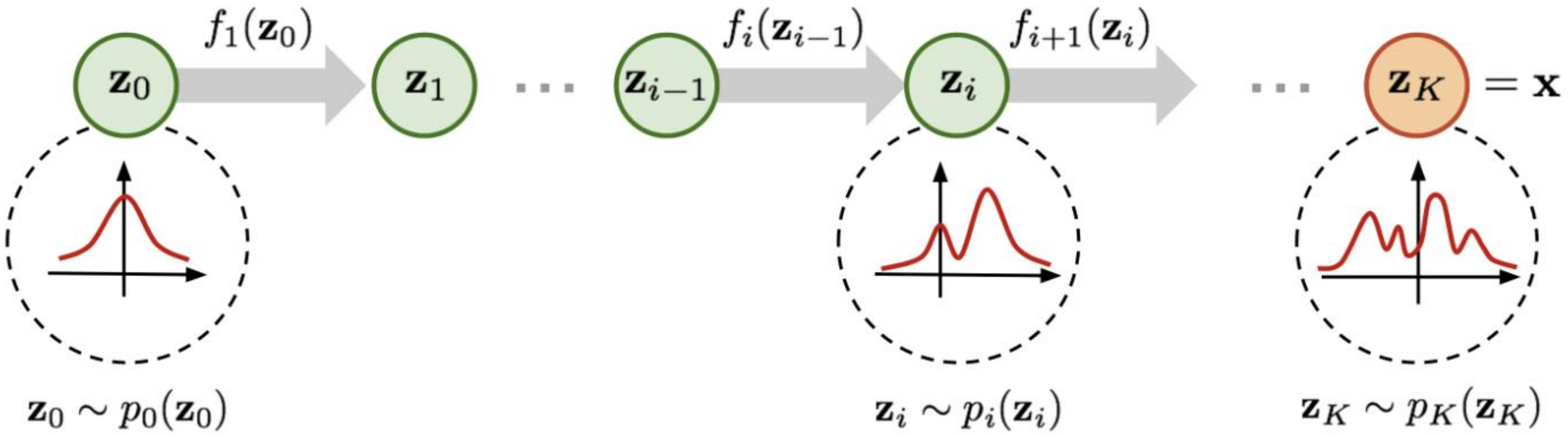
Pros and Cons

- (Discrete) Normalizing Flows
  - Different parameters at different steps
  - Restricted (invertible) architecture
- Continuous Normalizing Flows
  - Same parameters at different steps
  - Unrestricted architecture

Small f (e.g., single layer) -> not expressive

Large f (e.g., a large network) -> slow training

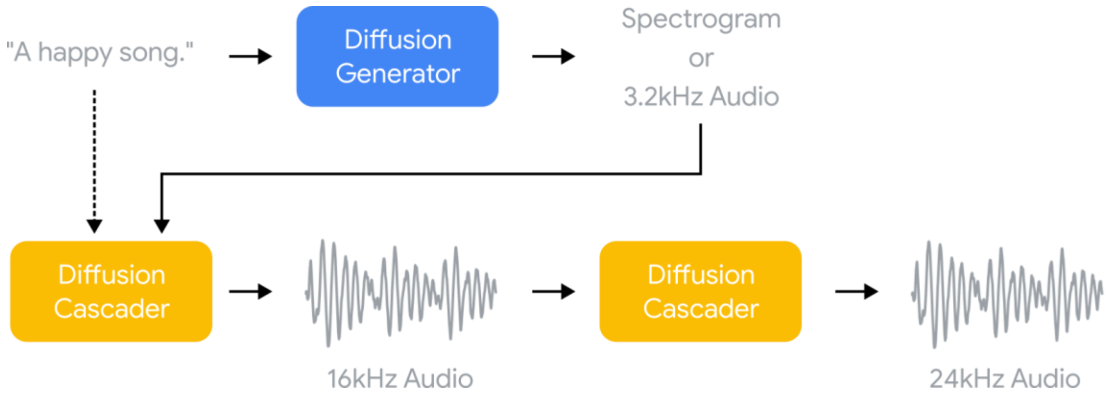
Diffusion is a CNF at inference time!
(but trained in a more efficient way)



Diffusion Models

A group of students are sitting in the classroom. The lecturer writes "Deep Generative Models" on the blackboard.

Image Generation

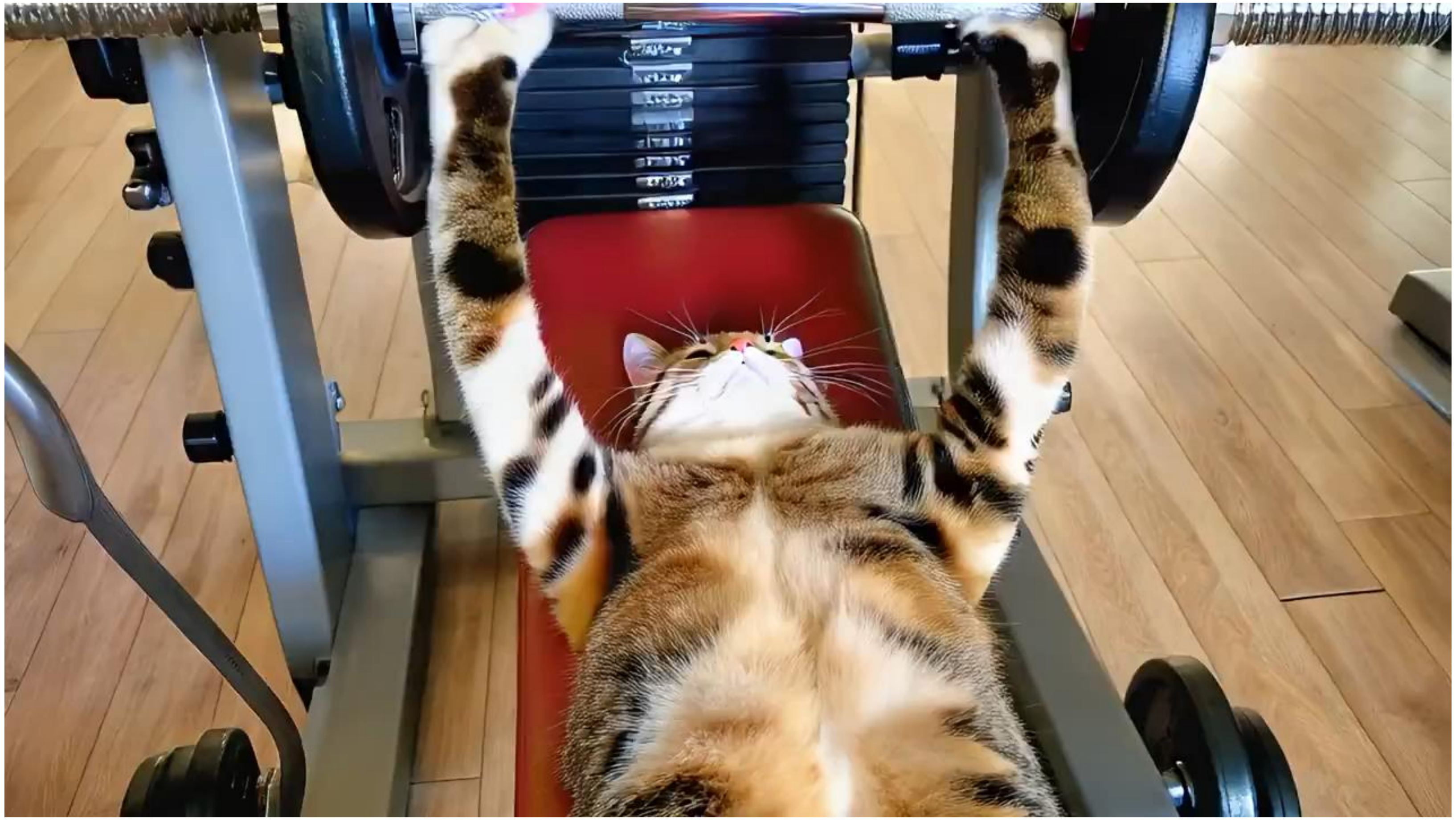


Audio Generation

Image generated by Flux

"Noise2Music: Text-conditioned Music Generation with Diffusion Models", Huang et al., 2023

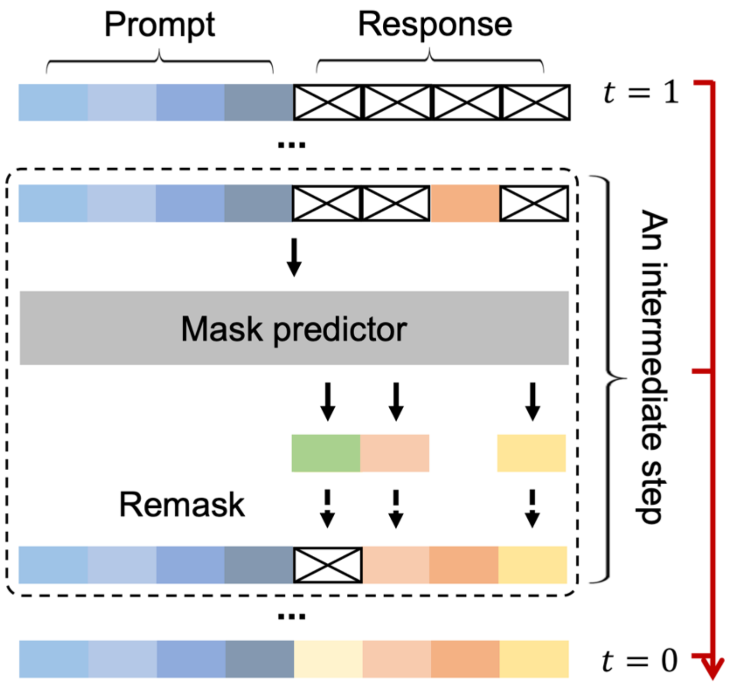
Diffusion Models – Video Generation

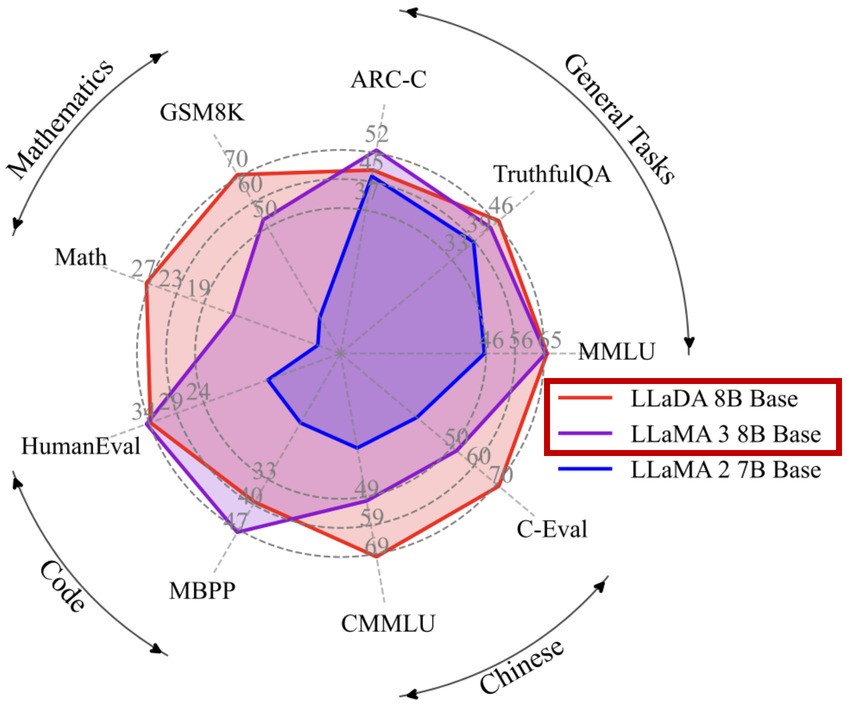


[Adobe Firefly Video]

Diffusion Models – Text Generation

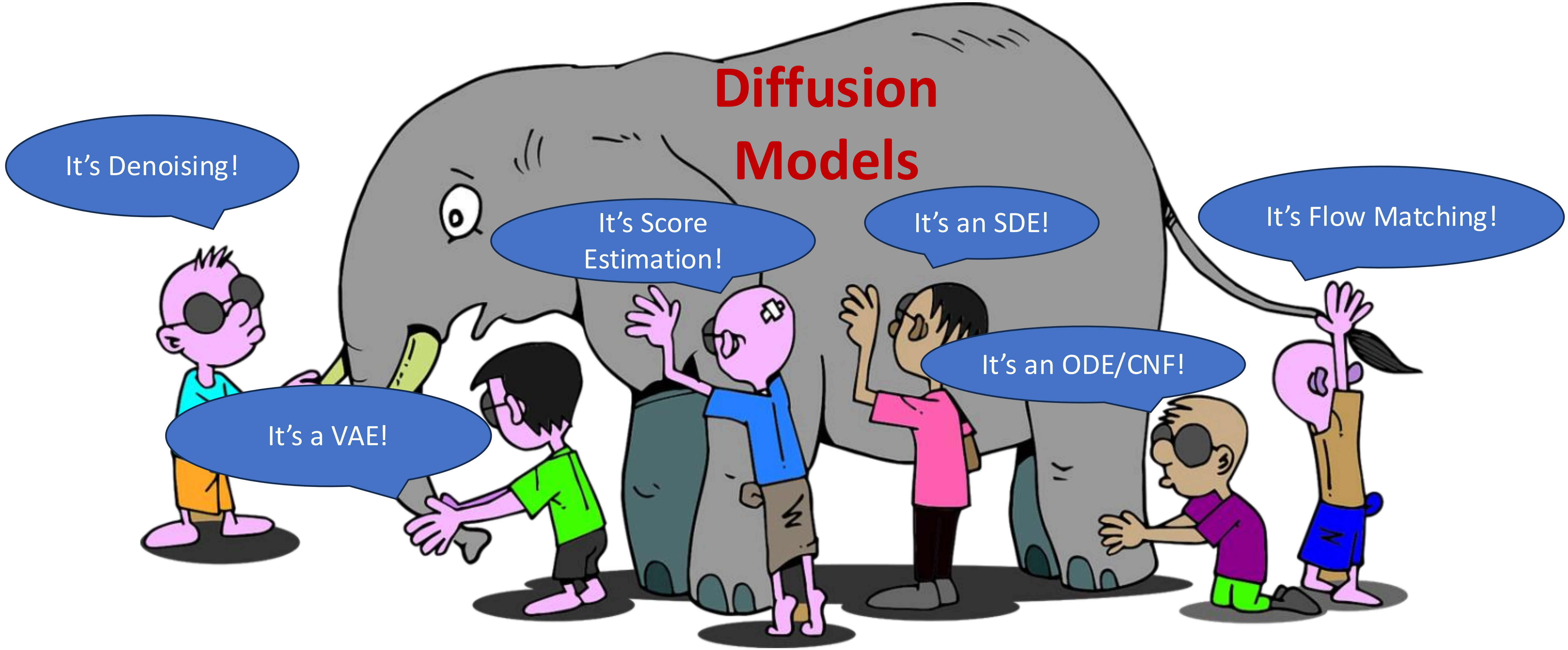
ball stone ice wash fish zoom



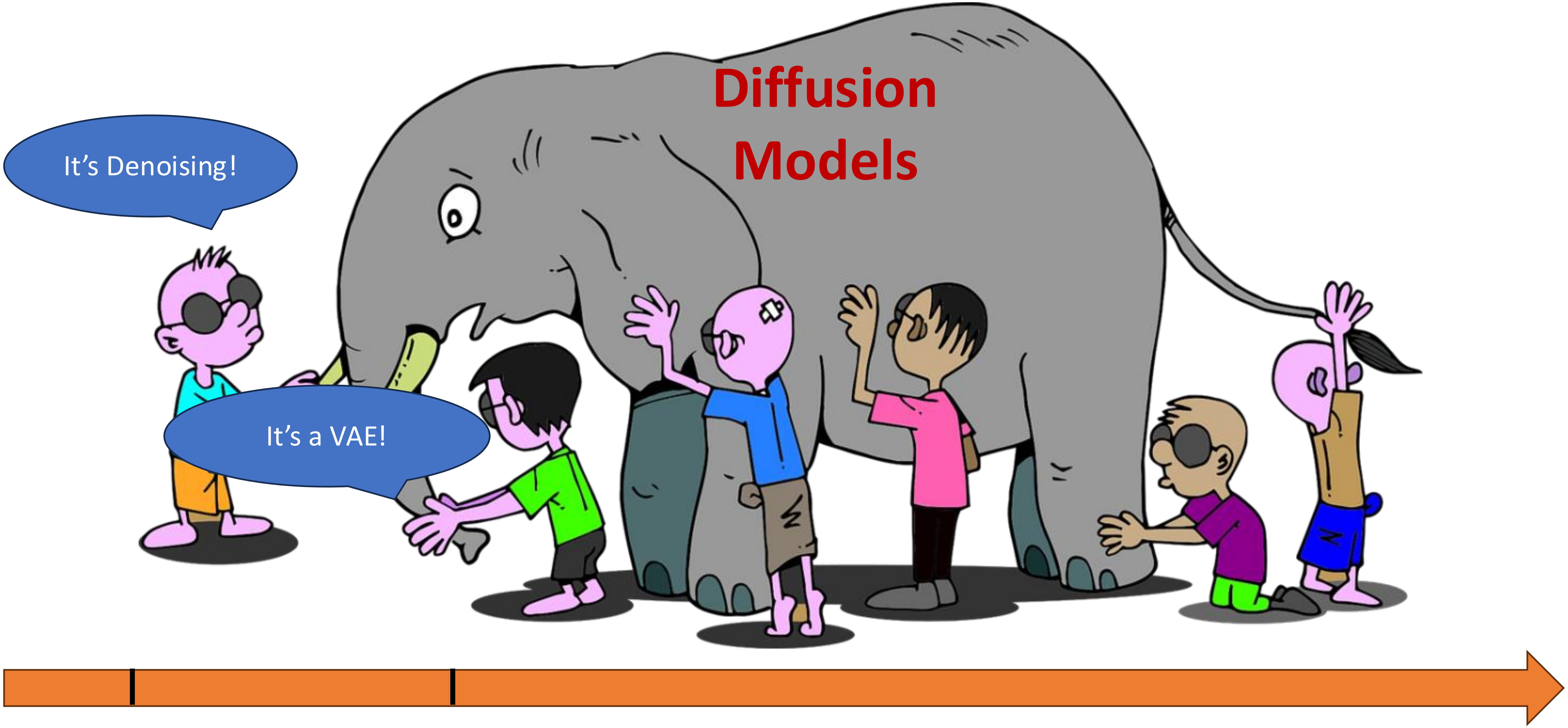


Text Generation

"Large Language Diffusion Models", Nie et al., 2025

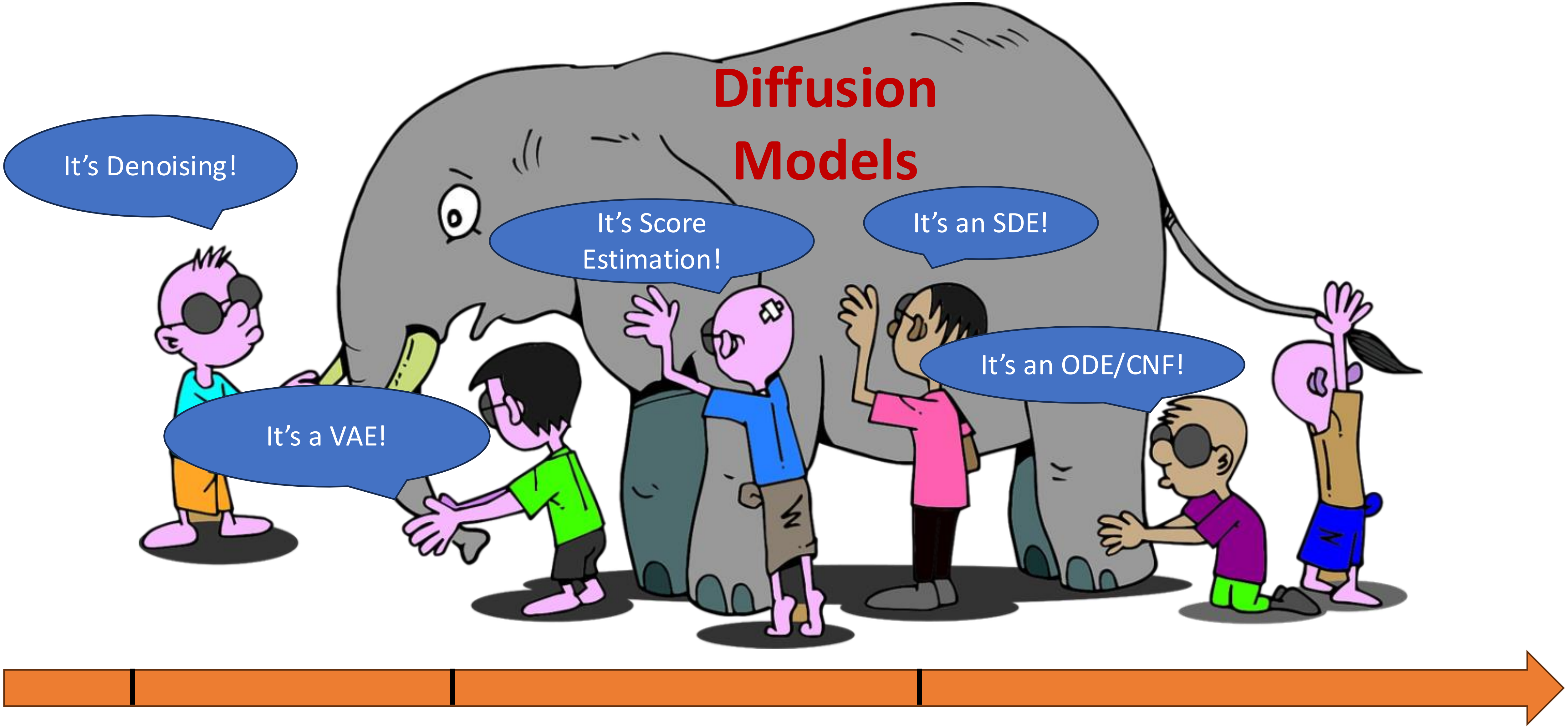


Perspectives on Diffusion Models



“Deep Unsupervised Learning using Nonequilibrium Thermodynamics” 2015

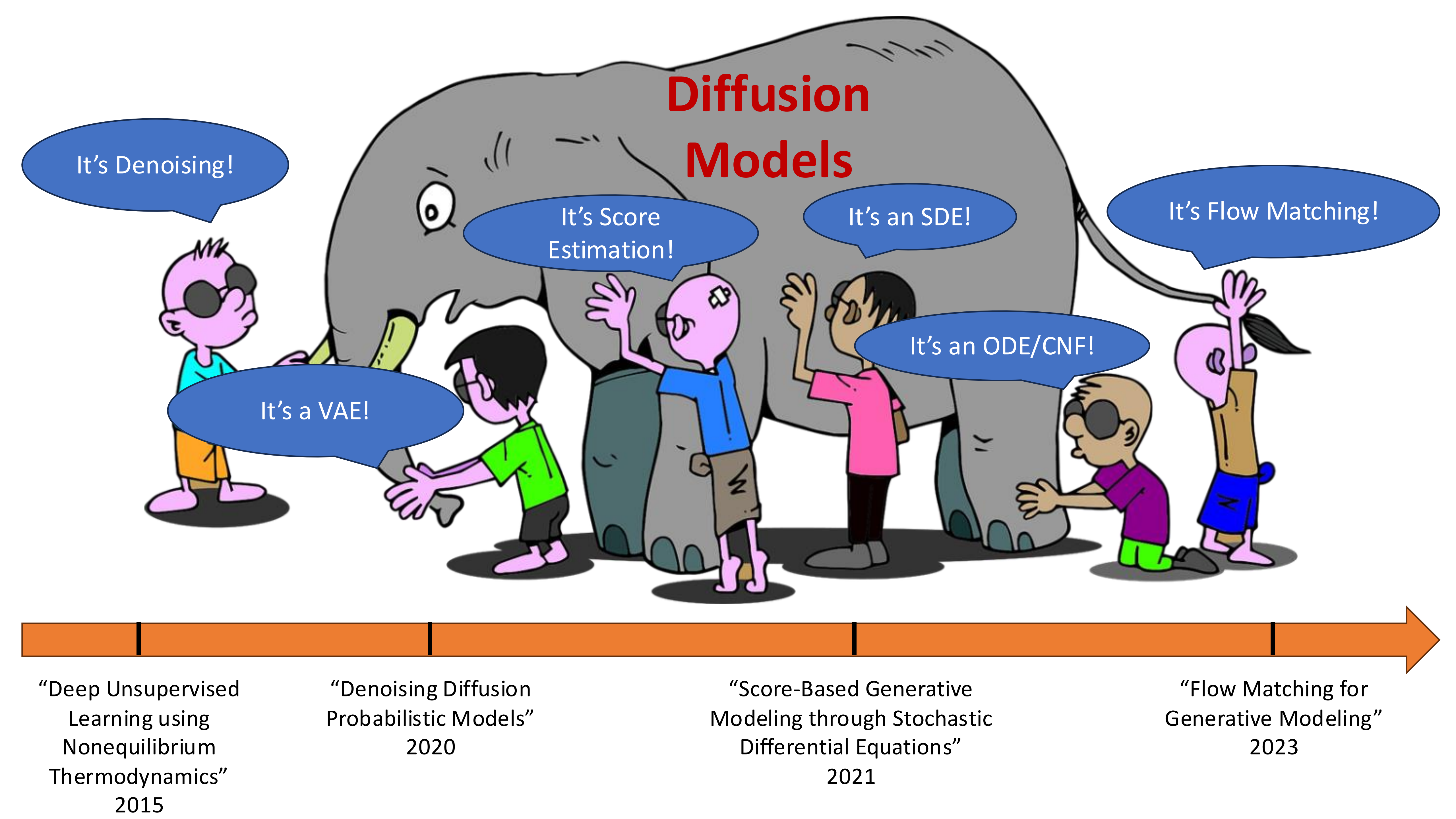
“Denoising Diffusion Probabilistic Models” 2020



“Deep Unsupervised Learning using Nonequilibrium Thermodynamics” 2015

“Denoising Diffusion Probabilistic Models” 2020

“Score-Based Generative Modeling through Stochastic Differential Equations” 2021


Diffusion Models
It’s Denoising!
It’s a VAE!
It’s Score Estimation!
It’s an SDE!
It’s an ODE/CNF!
It’s Flow Matching!
“Deep Unsupervised Learning using Nonequilibrium Thermodynamics” 2015
“Denoising Diffusion Probabilistic Models” 2020
“Score-Based Generative Modeling through Stochastic Differential Equations” 2021
“Flow Matching for Generative Modeling” 2023

Refresh: properties of Gaussian

Let $x \sim N(\mu_x, \sigma_x^2)$ and $y \sim N(\mu_y, \sigma_y^2)$ be two Gaussian random variables

- Sum of two Gaussians is a Gaussian

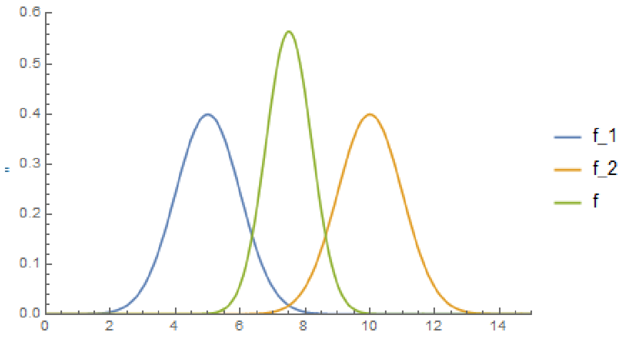
$$x + y \sim N(\mu_x + \mu_y\, , \sigma_x^2 + \sigma_y^2)$$

- KL Divergence between Gaussians

$$KL(p(x)|p(y)) = \frac{\sigma_x^2 + \left(\mu_x - \mu_y\right)^2}{2\sigma_y^2} + \log\frac{\sigma_y}{\sigma_x} - \frac{1}{2}$$

L2 distance between mean $$= \boxed{\frac{\left(\mu_x - \mu_y\right)^2}{2}} - \frac{1}{2} \text{ (if } \sigma_x = \sigma_y)$$

- Product of two Gaussians PDFs is a Gaussian
  - Why? Gaussian $\Leftrightarrow \log p(x)$ has quadratic form!



Diffusion Model is **Iterative Denoising**

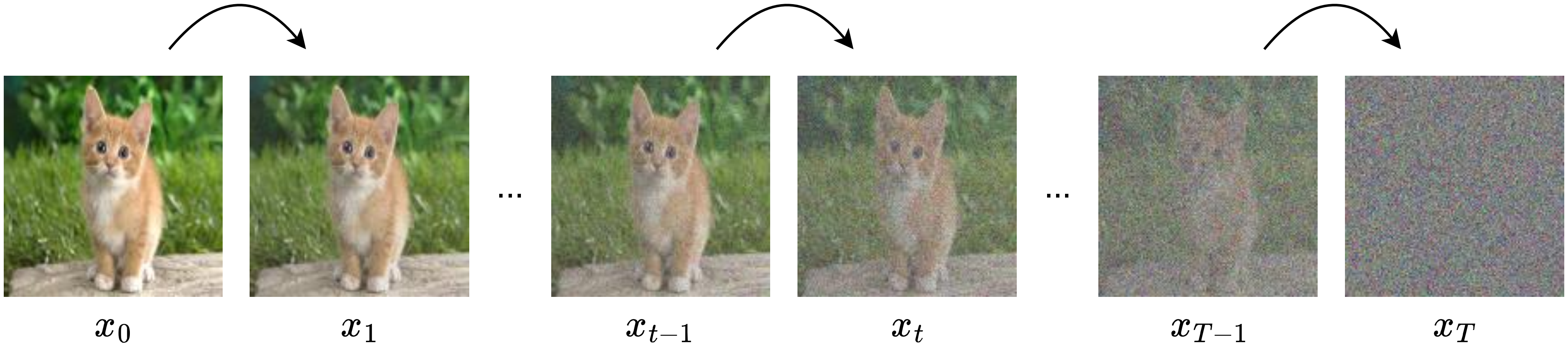
- Forward process (diffusion):
  - Iteratively inject Gaussian noise into clean data, until it's pure noise
  - Markovian: $q(x_{0:T}) = q(x_0) \prod_{t=1}^{T} q(x_t|x_{t-1})$

$$q(x_t|x_{t-1}) = N(\sqrt{1-\beta_t}x_{t-1}, \sqrt{\beta_t}I)$$

Reparameterization (like VAE): $x_t = \sqrt{1-\beta_t}x_{t-1} + \sqrt{\beta_t}\epsilon_{t-1}, \epsilon_{t-1} \sim N(0, I)$

β_t: Noise schedule

How much noise to add at each step?



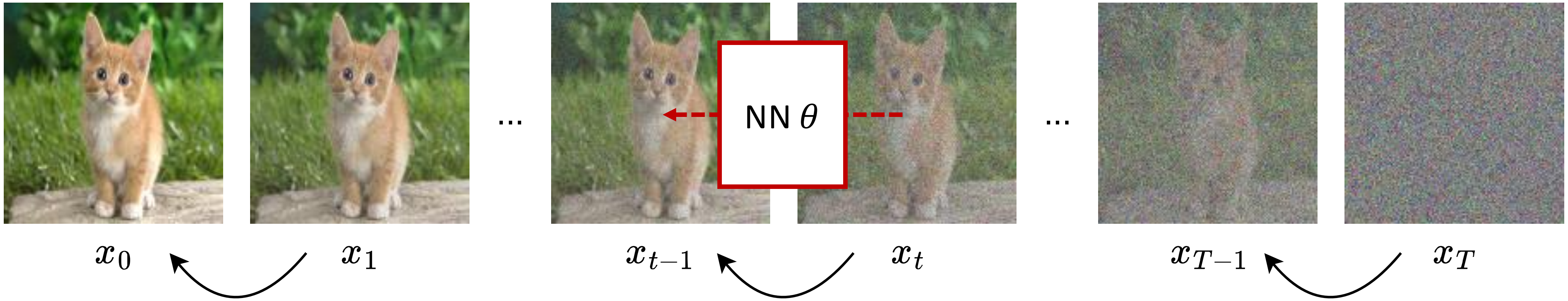
T is usually very large (e.g., 1000)

$q(x_T) = N(0, I)$ regardless of input

Diffusion Model is **Iterative Denoising**

- Forward process (diffusion):
  - Iteratively inject Gaussian noise into clean data, until it's pure noise
  - Markovian: $q(x_{0:T}) = q(x_0)\prod_{t=1}^{T} q(x_t|x_{t-1})$
- Reverse process (denoising):
  - Train a $\boxed{\text{model } \theta}$ to iteratively remove noise, starting from pure noise

A neural network!

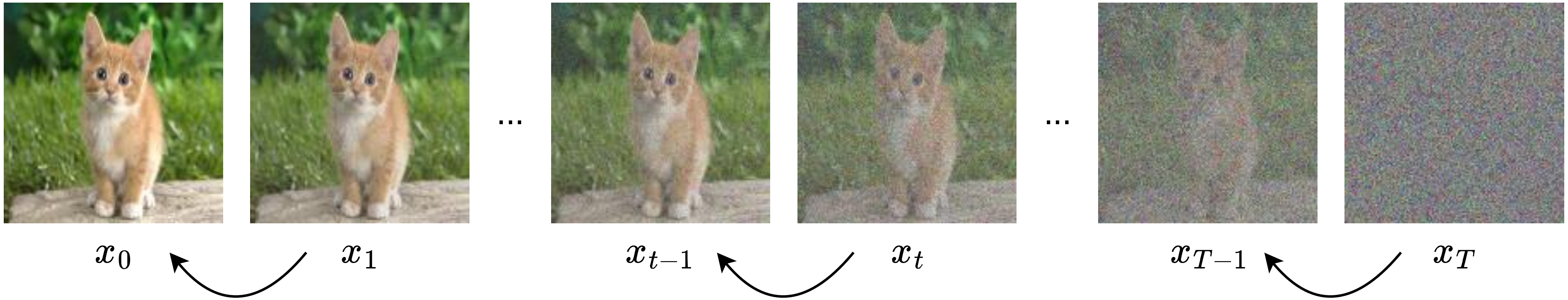


$$p_\theta(x_{t-1}|x_t) = N(\mu_\theta(x_t), \sigma_t^2 I)$$

Diffusion Model is **Iterative Denoising**

- Forward process (diffusion):
  - Iteratively inject Gaussian noise into clean data, until it's pure noise
  - Markovian: $q(x_{0:T}) = q(x_0)\prod_{t=1}^{T} q(x_t|x_{t-1})$
- Reverse process (denoising):
  - Train a model θ to iteratively remove noise, starting from pure noise
  - Markovian: $p_\theta(x_{0:T}) = p(x_T)\prod_{t=1}^{T} p_\theta(x_{t-1}|x_t)$

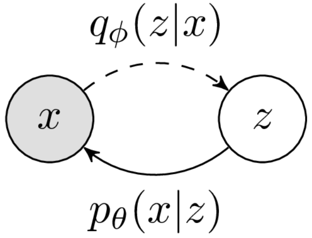
How to train it?

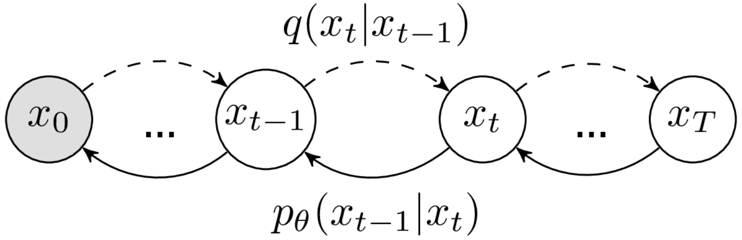


$$p_\theta(x_{t-1}|x_t) = N(\mu_\theta(x_t), \sigma_t^2 I)$$

Diffusion Model is a **VAE**

- ELBO in VAE: $\log p_\theta(x) \geq \mathbb{E}_{z\sim q_\phi(z|x)}[\log p_\theta(x|z)] - KL\left(q_\phi(z|x) \parallel p(z)\right)$
  $= \mathbb{E}_{z\sim q_\phi(z|x)} \log \frac{p_\theta(x,z)}{q_\phi(z|x)}$
- Assume the latent consists of **all** noisy images: $z = (x_1, x_2, \ldots, x_T)$
  - $\log p_\theta(x_0) \geq \mathbb{E}_{z\sim q} \log \frac{p_\theta(x_0, x_1, x_2, \ldots, x_T)}{q(x_1, x_2, \ldots, x_T|x_0)}$
    $= \mathbb{E}_{z\sim q} \log \frac{p_\theta(x_0|x_1)p_\theta(x_1|x_2)\ldots p_\theta(x_{T-1}|x_T)p_\theta(x_T)}{q(x_1|x_0)q(x_2|x_1)\ldots q(x_T|x_{T-1})}$ (Markovian)
    $= \mathbb{E}_{z\sim q} \log p_\theta(x_T) + \log \prod_{t=2}^{T}\left(\frac{p_\theta(x_{t-1}|x_t)}{q(x_t|x_{t-1})}\right) + \log\left(\frac{p_\theta(x_0|x_1)}{q(x_1|x_0)}\right)$





Diffusion Model is a **VAE**

- $\log p_\theta(x_0) \geq \mathbb{E}_{z\sim q} \log p_\theta(x_T) + \log \prod_{t=2}^{T} \left(\frac{p_\theta(x_{t-1}|x_t)}{q(x_t|x_{t-1})}\right) + \log\left(\frac{p_\theta(x_0|x_1)}{q(x_1|x_0)}\right)$

Diffusion Model is a **VAE**

$\tilde{x}_{t-1} = \frac{1}{\sqrt{\alpha_t}}\left(x_t - \frac{\beta_t}{\sqrt{1-\bar{\alpha}_t}}\epsilon\right)$ is the mean of $q(x_{t-1}|x_t, x_0)$

σ_t: standard deviation of $q(x_{t-1}|x_t, x_0)$

- $-\log p_\theta(x_0) \le \underbrace{\cancel{KL(q(x_T|x_0)||p(x_T))}}_{\text{Constant}} + \underbrace{\sum_{t>1} KL(q(x_{t-1}|x_t, x_0)||p_\theta(x_{t-1}|x_t))}_{\text{L2 distance}} - \underbrace{E_q[\log p_\theta(x_0|x_1)]}_{\text{L2 distance}}$

- $q(x_{t-1}|x_t, x_0)$ is a Gaussian distribution!

$\boxed{\text{Product of two Gaussians PDFs is a Gaussian!}}$

- $p_\theta(x_{t-1}|x_t) = N(\mu_\theta(x_t), \sigma_t^2 I)$ is also Gaussian!

$\boxed{\text{KL divergence between Gaussians = L2 distance between mean!}}$

- Loss: predict x_{t-1} from x_t

$$L = \frac{1}{2\sigma_t^2}\sum_{t=1}^{T}||\mu_\theta(x_{t-1}|x_t) - \tilde{x}_{t-1}||^2$$

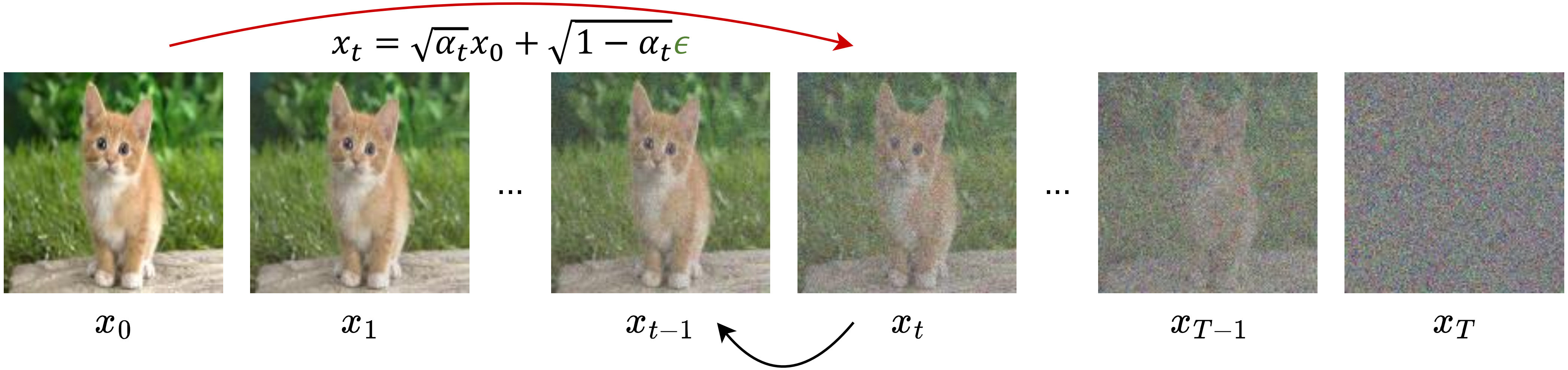
T is very large (e.g., 1000)… Do we have to compute 1000 loss terms in one training iteration? **NO!**

Training: One step at a time

- Add Gaussian noise **many** times = Add **one** (larger) Gaussian noise. Why?

Sum of two Gaussians random variables is still a Gaussian!

- $q(x_t|x_{t-1}) = N(\sqrt{1-\beta_t}x_{t-1}, \sqrt{\beta_t}I) \Longrightarrow q(x_t|x_0) = (\sqrt{\alpha_t}x_0, \sqrt{1-\alpha_t}I)$
  - $\alpha_t = \prod_{s=1}^{t}(1-\beta_s)$: How much of the signal still remains?

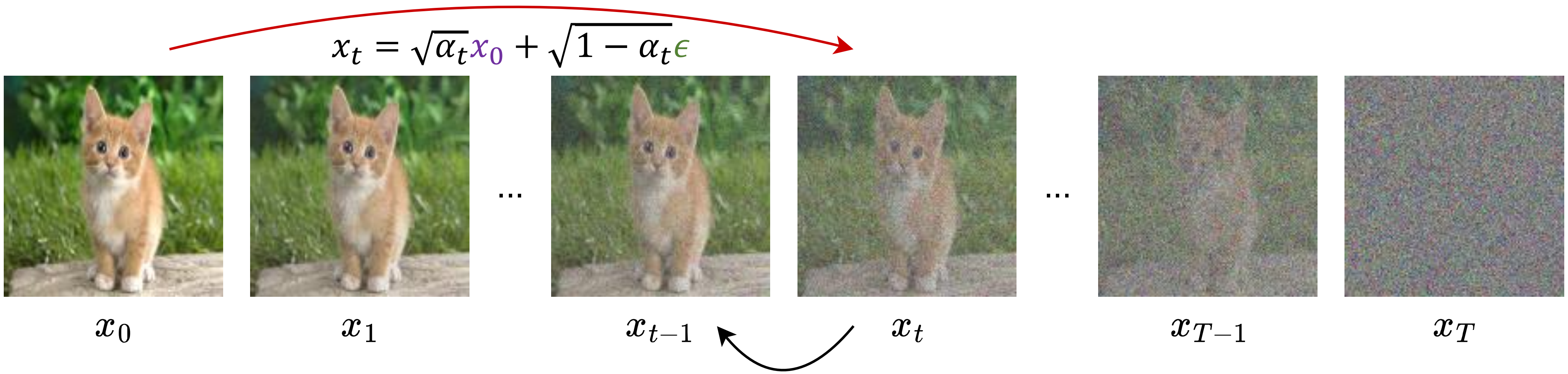


Three Equivalent Prediction Targets

- $\begin{cases} \tilde{x}_{t-1} = \frac{1}{\sqrt{1-\beta_t}}\left(x_t - \frac{\beta_t}{\sqrt{1-\alpha_t}}\epsilon\right) \\ x_t = \sqrt{\alpha_t}x_0 + \sqrt{1-\alpha_t}\epsilon \end{cases}$ 2 equations, 3 unknown variables (x_t is known)
- All below targets are **equivalent**:
  - The slightly less noisy image: $\tilde{x}_{t-1}$
  - Clean image: x_0
  - The added noise: ϵ Anything else?
  - Any linear combination of the above three (e.g., $x_0 - \epsilon$)!
- In other words, the below loss functions are the same (with different $\underbrace{w(t), w'(t), w''(t)}_{\text{Time-dependent weighting}}$)
  - $\sum_{t=1}^{T} w(t)\|\mu_\theta(x_{t-1}|x_t) - \tilde{x}_{t-1}\|^2$
  - $\sum_{t=1}^{T} w'(t)\|\epsilon_\theta(x_t) - \epsilon\|^2$
  - $\sum_{t=1}^{T} w''(t)\|\hat{x}_0(x_t;\theta) - x_0\|^2$

Three Equivalent Prediction Targets

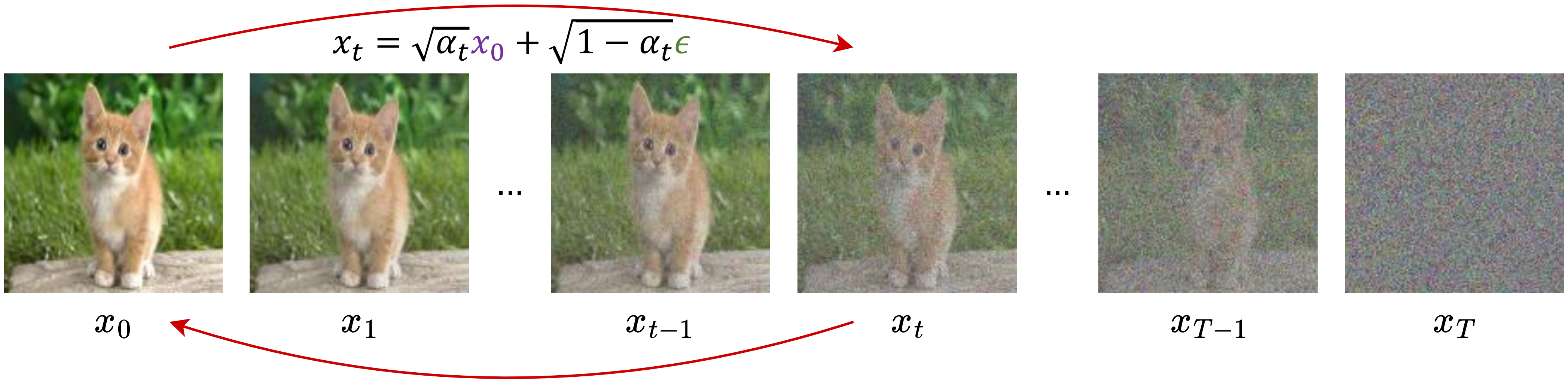
- All below targets are **equivalent**:
  - **The slightly less noisy image: $\tilde{\boldsymbol{x}}_{\boldsymbol{t-1}}$**
  - Clean image: x_0
  - The added noise: ϵ



Three Equivalent Prediction Targets

- All below targets are **equivalent**:
  - The slightly less noisy image: $\tilde{x}_{t-1}$
  - **Clean image: x_0**
  - The added noise: ϵ

$$L = \mathbb{E}_{x_0,\, t,\, \epsilon}\, w(t) \| \hat{x}_0(x_t; \theta) - x_0 \|^2$$

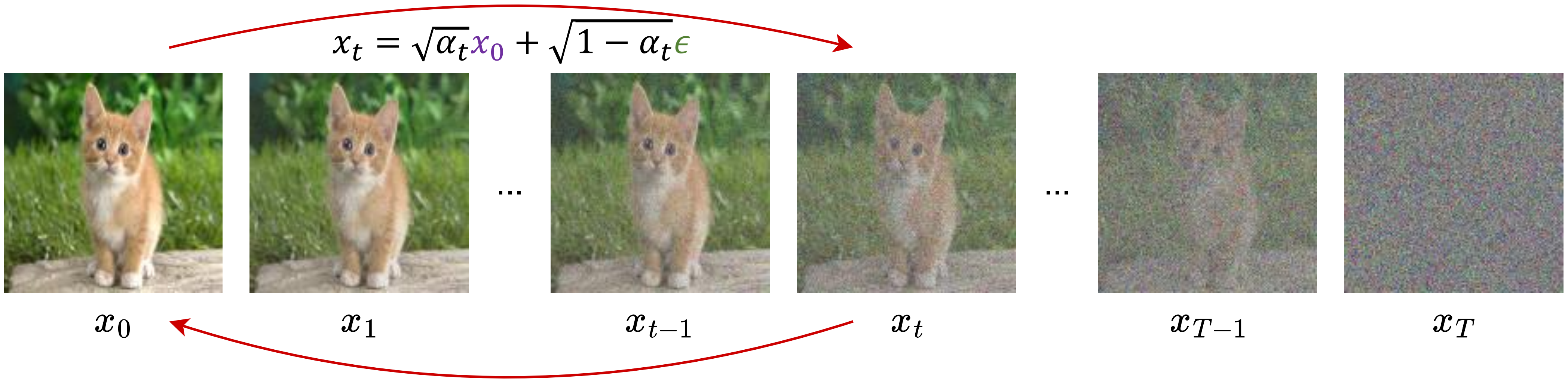


Three Equivalent Prediction Targets

- All below targets are **equivalent**:
  - The slightly less noisy image: $\tilde{x}_{t-1}$
  - Clean image: x_0
  - **The added noise: ϵ**

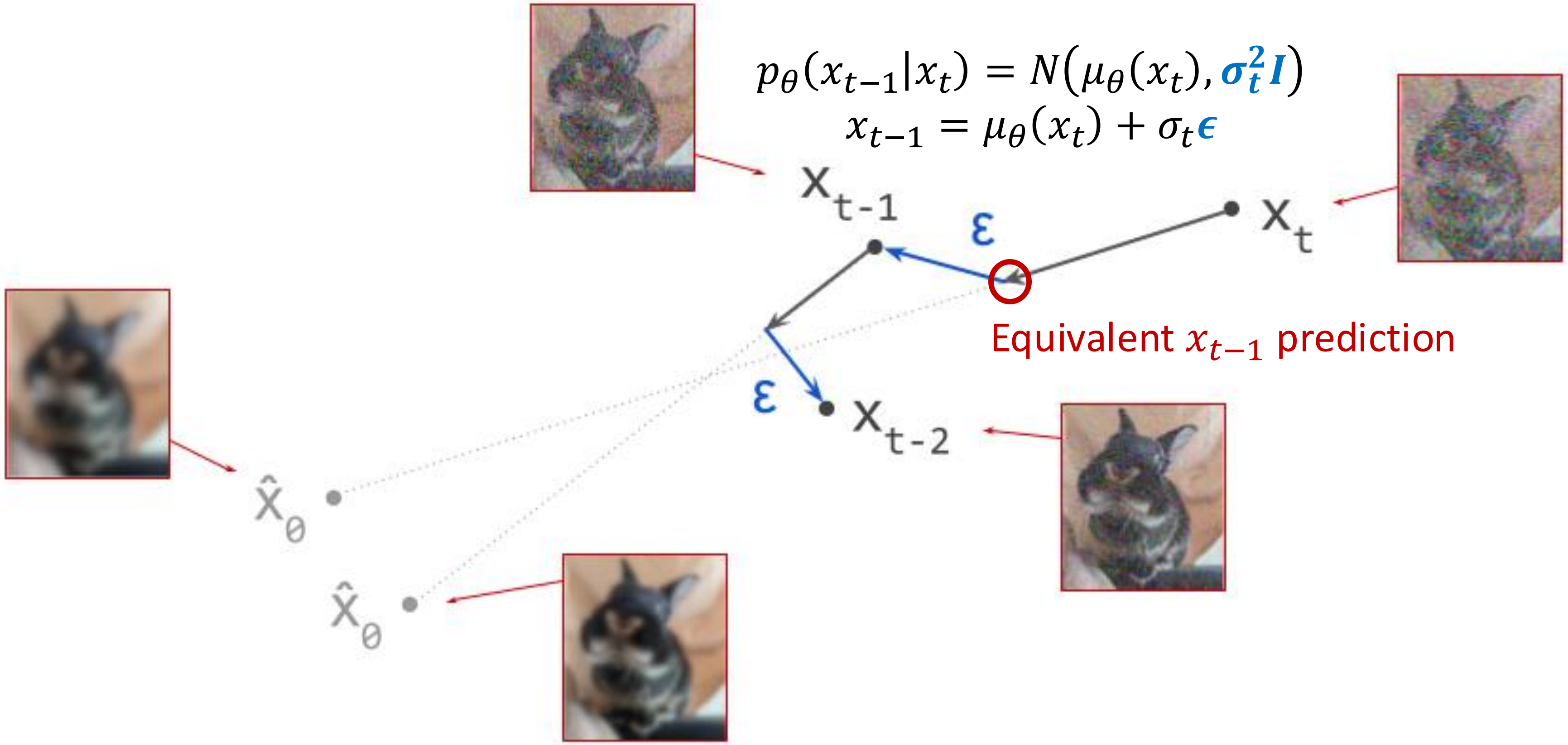
Convert back to $\tilde{x}_{t-1}$ prediction during sampling!

$$L = \mathbb{E}_{x_0,\, t,\, \epsilon}\, w'(t) \|\epsilon_\theta(x_t) - \epsilon\|^2$$



Sampling

$$p_\theta(x_{t-1}|x_t) = N\left(\mu_\theta(x_t), \boldsymbol{\sigma_t^2 I}\right)$$
$$x_{t-1} = \mu_\theta(x_t) + \sigma_t \boldsymbol{\epsilon}$$



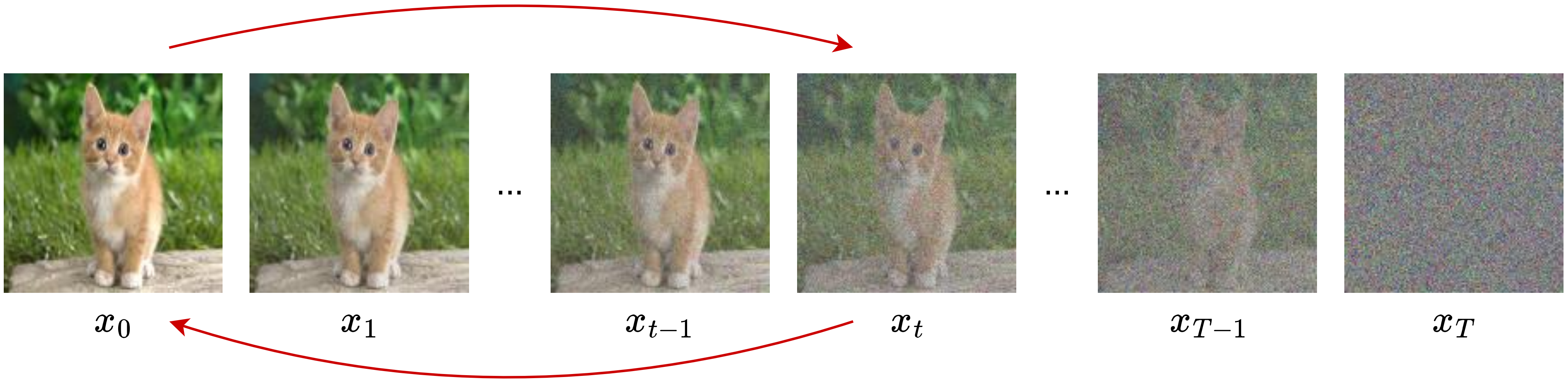
repeat T times…

repeat

Sander Dieleman, “The geometry of diffusion guidance”

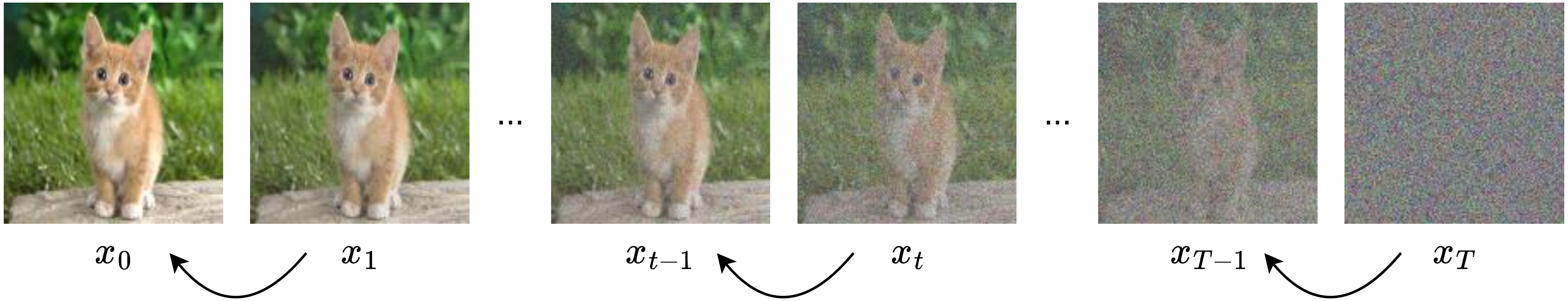
Recap: Diffusion Model is a Denoising VAE

- Training: Denoising objective



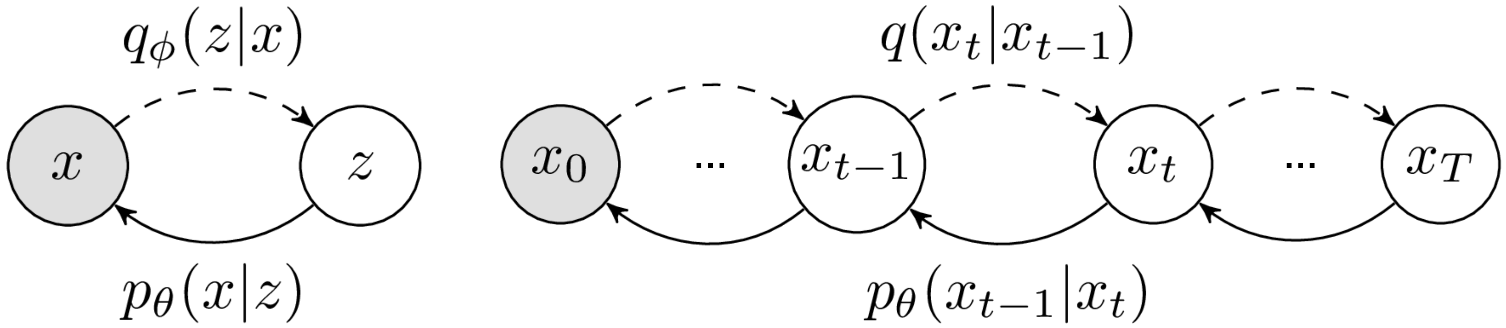
Recap: Diffusion Model is a Denoising VAE

- Training: Denoising objective
- Inference: Starting from pure noise, iteratively remove noise



Recap: Diffusion Model is a Denoising VAE

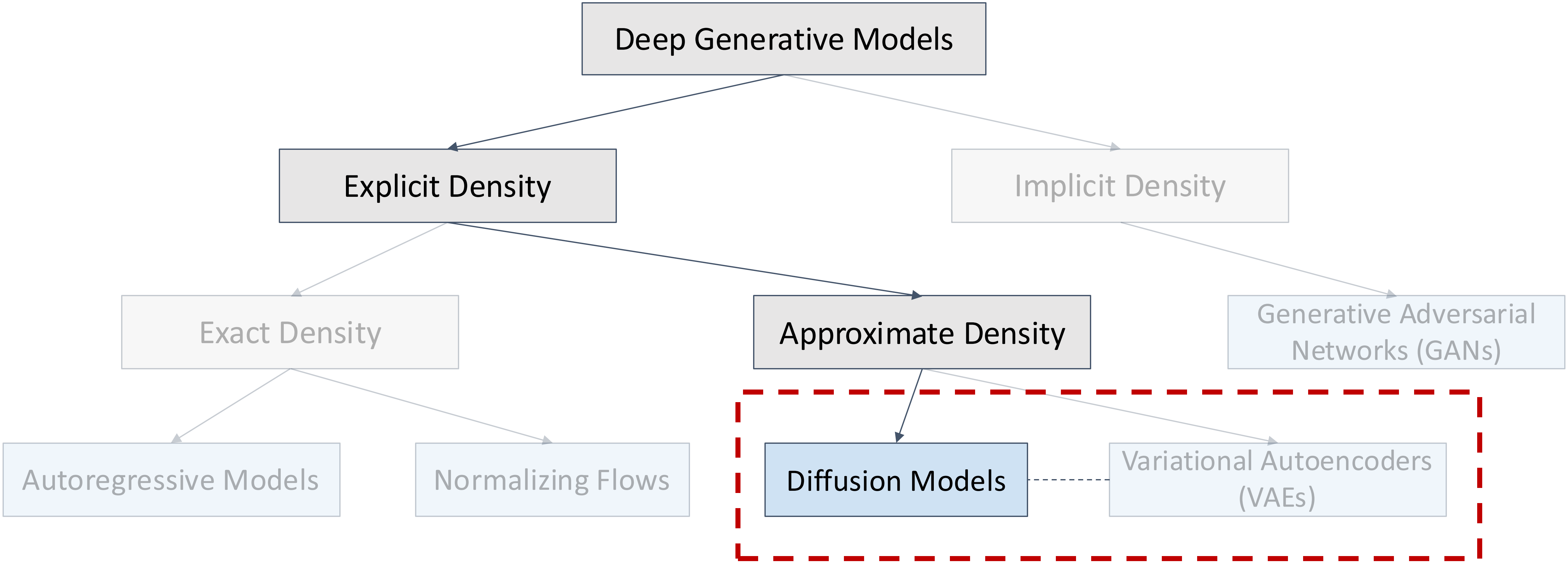
- Training: Denoising objective
- Inference: Starting from pure noise, iteratively remove noise
- Connection to VAE
  - A Hierarchical VAE with a fixed encoder (so less expressive), BUT:
  - Much easier to optimize (just one level at each iteration)



Recap: Diffusion Model is a Denoising VAE

- Training: Denoising objective
- Inference: Starting from pure noise, iteratively remove noise
- Connection to VAE
  - A Hierarchical VAE with a fixed encoder (so less expressive), BUT:
  - Much easier to optimize (just one level at each iteration)
- Three equivalent prediction targets
  - $\tilde{x}_{t-1}, x_0, \epsilon$

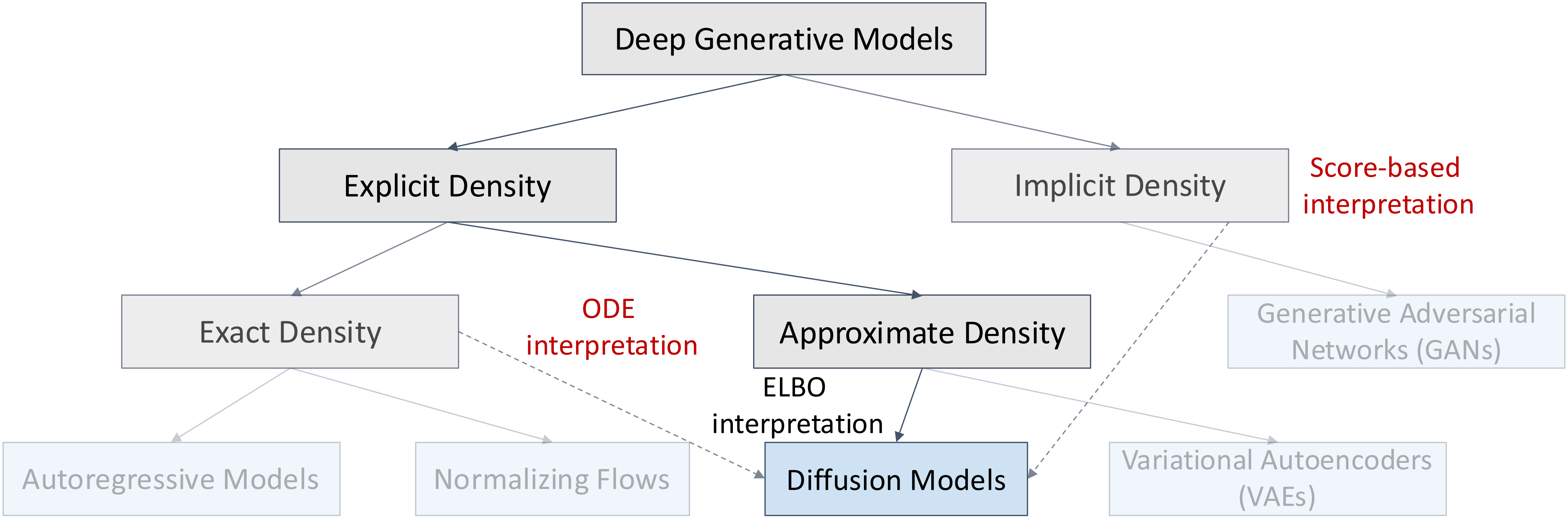
Diffusion Models



Maximizing ELBO (Evidence Lower Bound)

Is it the full story?

Diffusion Models (continue...)



5 Minute Quiz

- On Canvas
- Passcode: crocodile